10 things you can do to create better documentation

Version 1.0
July 13, 2007

By Alan Norton

Takeaway

Documentation isn't exactly the reason why most IT professional chose their career. But with the right amount of enthusiasm, hard work and attention to detail you can become a much more valuable asset to your employer and enhance your resume. While the ten tips in this blog post are mainly targeted toward the latter category, managers and supervisors can also benefit from them.

Let's be honest. No one wants to write about it, read about it or actually have to do it. I'm talking about documentation, usually defined as the second to last step in the Project Life Cycle. Although documentation can and should occur at any point in the Project Life Cycle, I will be focusing my tips on the documentation phase. Please note that there are two subsets of documentation I will be discussing, *End User Documentation* and *System or Internal Documentation*.

The Project Life Cycle

While doing the research for this article, I pulled out the *Encyclopedia of Computer Science, Editors Anthony Ralson and Chester L, Meek, Pretrocelli and Charter 1976.* It is definitely a first edition classic, but some things are timeless. It states in the Documentation listing:

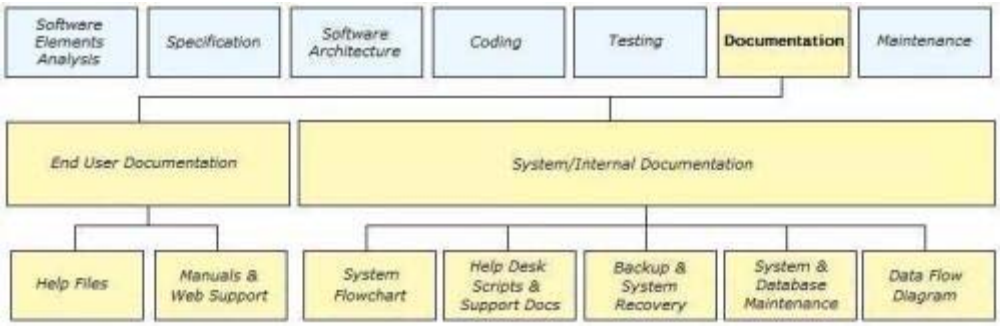
> *"Documentation is a vital part of developing and using a computer-based system. In some commercial organizations, 20 to 40% of the total development effort goes into the documentation of the new system, recording how the new system is to work and how it was developed." Author, K.R. London*

I was curious if that was still true today, and if so, why documentation is not given the media coverage it deserves. I could not find a lot of information on the costs of documentation during the Project Life Cycle. (See **Figure A**) I suspect that these costs are often understated or not properly collected. For IT professionals whose job is to develop working systems, many may consider admitting the actual costs of documentation as embarrassing or reflecting poorly on their productivity and work.

Your opinion of the documentation phase of the Project Life Cycle likely depends on your role in the IT organization. If you are a manager or project lead, documentation is critical to the success of the project. If you are in a non-managerial position, documentation is an annoying nuisance that gets in the way of your *real* work. While the ten tips below are mainly targeted toward the latter category, managers and supervisors can also benefit from them.

Copyright ©2007 CNET Networks, Inc. All rights reserved.
For more downloads and a free TechRepublic membership, please visit http://downloads.techrepublic.com.

Figure A



Source for the seven Project Life Cycle steps: WikipediA

1. Document with pictures if possible

The old adage a picture is worth a thousand words means that by using pictures to augment your text you can minimize the length and complexity of your documentation. System users like having pictures, diagrams, tables and bulleted lists for quick reference.

2. Give examples

Examples are an excellent way for end users to quickly grasp concepts that they may not fully understand. It is also a good way for an end user learning new software to sit down and tackle a new challenge more easily. Here is an example of an example with pictures:

Previous Versions in Vista Business, Ultimate and Enterprise

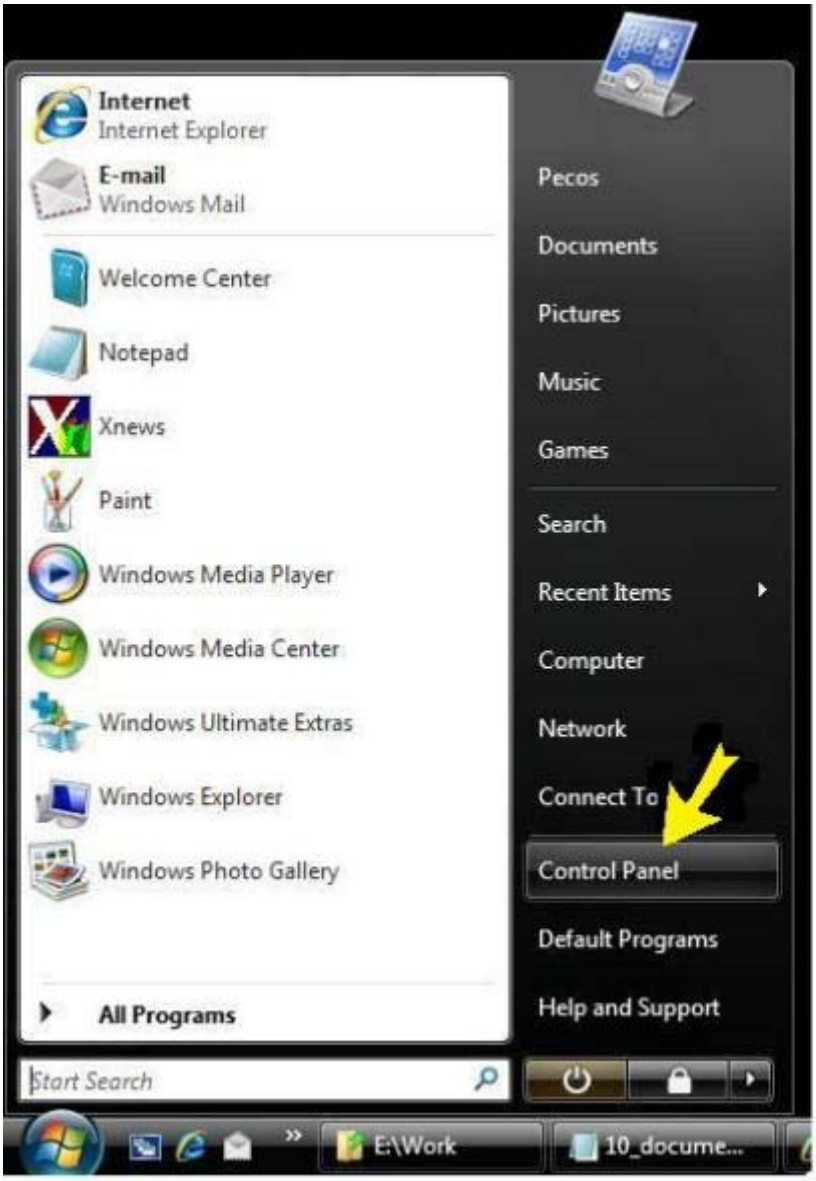
Previous Versions is the term that Microsoft uses for saving shadow copies of a file. If you are working on a document or any other project that is labor intensive and have accidentally lost part or all of your work, you can go back to a previous version. Think of it as an automated way to make snapshots of how the file looked at previous points in time.

But first you will have to configure Vista so that Previous Versions (Figures B through E) will be turned on *for the logical drives/partitions that you want the ability to restore the previous versions of your files.*

Enabling Previous Versions

Copyright ©2006 CNET Networks, Inc. All rights reserved.
For more downloads and a free TechRepublic membership, please visit http://downloads.techrepublic.com.

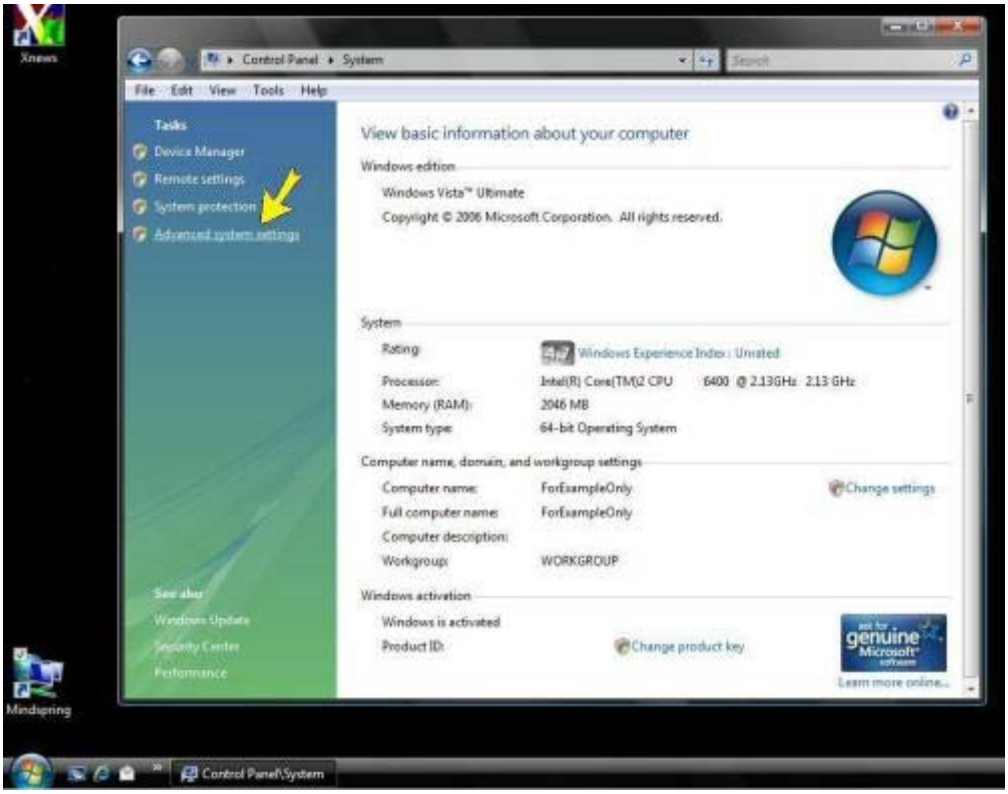
Figure B



Select Start | Control Panel

Copyright ©2006 CNET Networks, Inc. All rights reserved.
For more downloads and a free TechRepublic membership, please visit http://downloads.techrepublic.com.

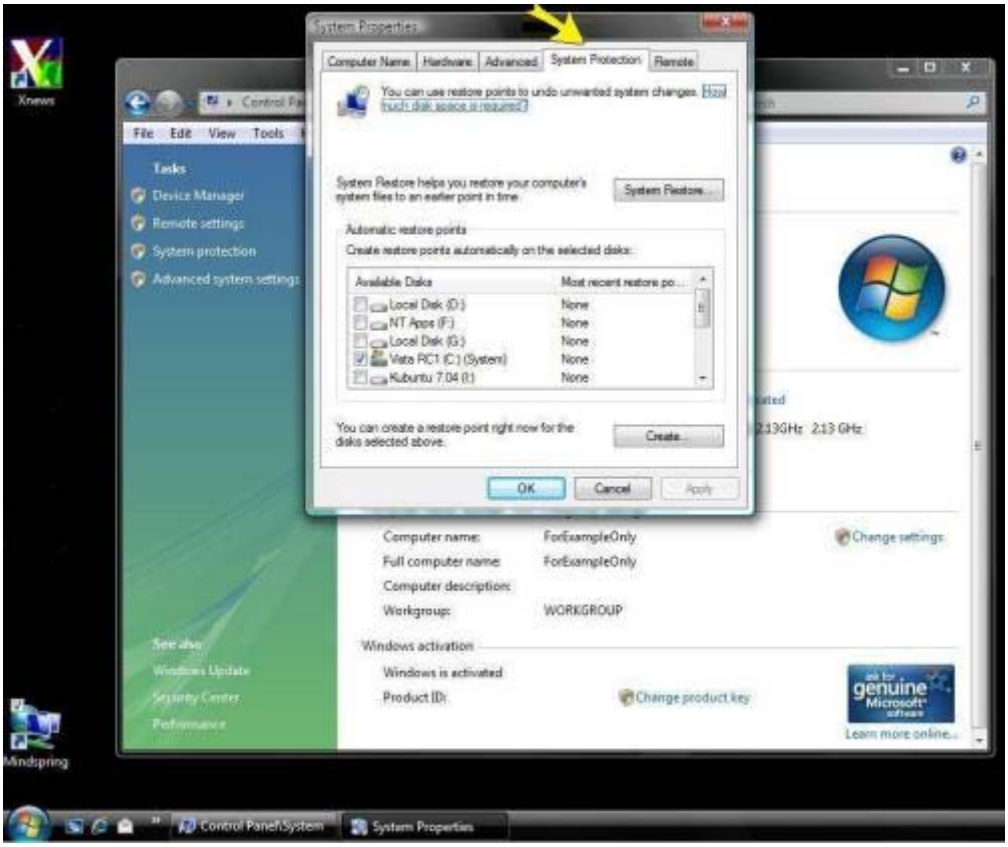
Figure C



Left click Advanced system settings -- Left click Continue if prompted by the UAC (User Account Control)

Copyright ©2006 CNET Networks, Inc. All rights reserved.
For more downloads and a free TechRepublic membership, please visit http://downloads.techrepublic.com.

Figure D

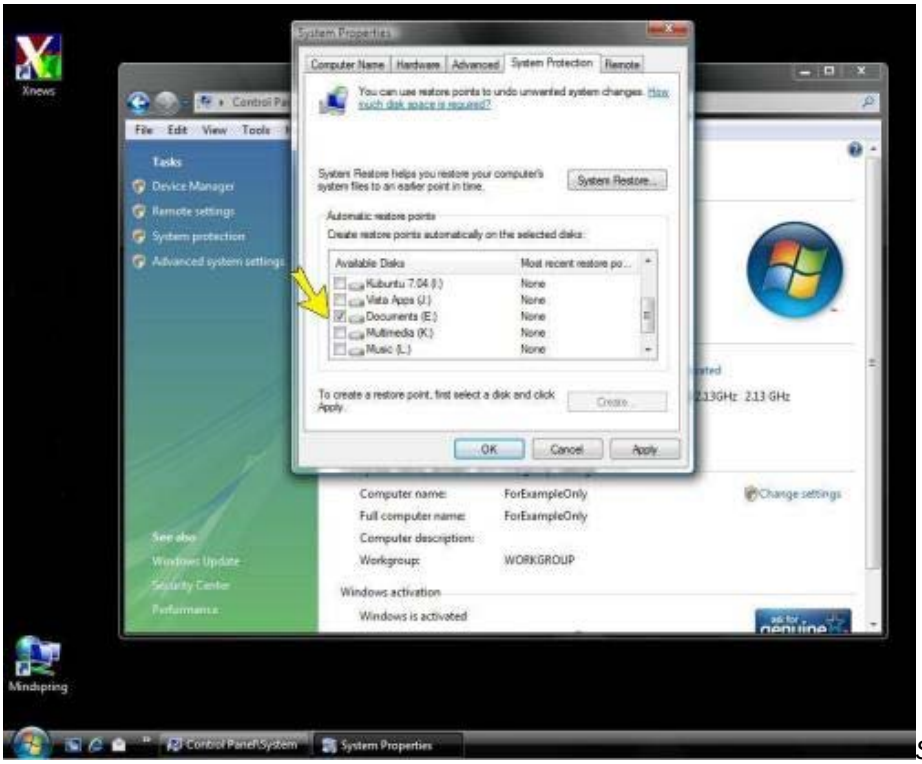


Left click on the System Protection tab

NOTE: *The Vista logical drive/partition is already enabled by default. Change this setting only if you fully understand the consequences of doing so.*

Copyright ©2006 CNET Networks, Inc. All rights reserved.
For more downloads and a free TechRepublic membership, please visit http://downloads.techrepublic.com.

Figure E



Scroll down and check logical drive

Left Click on the scroll bar and scroll down through the logical drives/partitions until you find the one that you want to enable Previous Versions. That logical drive name is Documents in this example. Left Click the checkbox next to the logical drive name.

NOTE: Shadow copies are created based on the time and frequency of system created restore points. According to the Vista help file, this is typically once a day.

Left click Apply and then left click OK to close the System Properties window. Left click File | Close or left click on the red X box in the upper right part of the Control Panel | System window to close it.

To restore the previous version of a file you can right click on the file name in Explorer and left click Restore previous versions

NOTE: Once a shadow copy of a file has been restored, it is no longer available for a second restore. Another shadow copy will not be created until the next system restore point is created. This means that any files saved before the next system restore point cannot be restored to the same previous version you used previously. Caution should be exercised saving files after using the Restore Previous Versions option until the next restore point occurs and another shadow copy can be created.

3. Don't presume to assume

Even if you know your targeted user base, your documentation needs to be written so that anyone with only basic computer skills can read it and learn how to properly use the system. Step by step instructions should be provided when possible but consider placing them in an appendix, a separate chapter or available via a hyperlink to avoid clutter. If you are doing the documentation, change your mindset so that you place yourself in the shoes of a new system user. That can be difficult to do at first, but if you pay attention to details and fully document all features and functions, you can create documentation that doesn't assume that the user can figure out information and procedures that you have failed to include.

Don't assume that your end user understands all of those acronyms that litter the IT landscape. The first time you present a new acronym, detail exactly what the acronym stands for.

Copyright ©2006 CNET Networks, Inc. All rights reserved.
For more downloads and a free TechRepublic membership, please visit http://downloads.techrepublic.com.

4. Anticipate problems

When testing your system you should have tried your best to break the software any way you could. If your software has known issues (developers like to call them issues, end users call them bugs), document a workaround and provide it to your users and the help desk. You will not only save a lot of frustration for the end users but also a lot of extra calls to the help desk.

Document the events that are inevitable during the lifetime of any long-lived system:

- What workarounds are available while the system or network is down?
- How do you recover from a server outage, a hard disk crash or database corruption?
- How does someone who knows absolutely nothing about your system get the system up and running again?

Your documentation should anticipate these problems and provide a detailed plan and instructions for system recovery.

Will the person who replaces you know where to find your documentation and any purchased vendor application documentation? All of these documents should be neatly organized and stored together in a safe and known place.

Another good example of anticipating problems is the Y2K Millennium Bug problem and solution. The media began reporting in the late 1990's that systems and software were likely to fail due to the storage of only two digits for the year in legacy systems. This problem was anticipated in advance and a lot of effort went into fixing the problem before it occurred. Software in development was built and certified as Y2K compliant years in advance of January 1, 2000. The results were remarkably successful. Except for a few minor reported problems, New Years day 2000 was a festive occasion and not a disaster for the IT community, though a lot of us were on-call *just in case*.

The same mindset can be used to anticipate problems that might arise in your documentation. The Y2K problem also illustrates the need for continual document updating. System/Internal documentation was changed to note the Y2K compliance or non-compliance of software and systems. For older legacy systems, workarounds were found and documented.

5. Test your documentation

Sit down and follow your own instructions. If you are documenting the building of a server, a network or any other IT system, start with a clean partition and build everything from scratch. You will undoubtedly discover that you have left something out or that some of your instructions are unclear.

Work with an uninformed but committed co-worker to get feedback before you publish. Let them test out your documentation.

You will be amazed at what you will learn when you sit a person down to work with your software and documentation for the first time. A lot of obvious features of the software to you will not be so obvious to someone who is honest and willing to work with you. Watch closely what your *guinea pig* does while navigating your software. Ask for feedback and take notes.

I remember the feedback I got during the testing of one of my projects. The feedback was written in an e-mail so I could review it point by point. The first thought that came to mind was "how long will this take to do?" You may also take these comments as critical or personal. Don't make that mistake. Looking back on it now, I should have implemented more of the missing features that my helpful critic had provided.

Use this opportunity to make final tweaks to your project. Feedback during the documentation process can help you make the overall project more successful.

I was writing a review for the Foxconn 975X7AB-8EKRS2H motherboard and I ran across two errors in the manual. I wasn't the first person to review the board. Foxconn had missed the errors and all of the other reviewers had missed the errors as well. One mistake in the manual was far from trivial.

The diagram in the manual showing the normal position of the clear CMOS jumper setting was incorrect. I know because when turning the motherboard over to verify the proper seating of the heatsink, the jumper fell off. I put the jumper back on according to the instructions in the manual. The computer failed to POST. After a careful look at the tiny diagram on the motherboard, I discovered the error and corrected the misplaced jumper.

Copyright ©2006 CNET Networks, Inc. All rights reserved.
For more downloads and a free TechRepublic membership, please visit http://downloads.techrepublic.com.

I was working with a tech from Foxconn at the time who was kind enough to answer my questions and I informed him of the error. Documentation errors like this are easy to miss and can lead to potentially large costs to the manufacturer. I would have missed the error myself were it not for the fact that the jumper had been loose enough to fall off when turning the motherboard over.

6. Humanize your work

How many times have you read a user manual and wondered if there really was a human at the other end of the creation of the manual - or was it a computer that made that manual? While you don't want to create a colorful novel, humanize the document just enough with some of your personality so that a reader will feel a little more comfortable while reading it.

7. Explore new technologies

Documentation can be costly even when done correctly. New technologies will continue to be created to help create more effective documentation that is less costly to develop. Look at these new tools as opportunities to reduce the time and cost of the documentation process.

Documenting as part of a project team can be especially difficult. Your documentation needs to be shared and added to the documentation of other team members. Changes have to be made, often on a daily basis. Software exists that will allow for this and will not only help to ensure a standardized end product it will also help to foster the sharing of ideas and knowledge among the team members.

While working at CSC (Computer Sciences Corporation) I had experimented with Microsoft's Agent and text to speech technology with mixed results. I always thought that it offered some wonderful ways to guide a new user through some of the features of my system. Some may remember that offensive little paper clip character with the blinking eyes in Word 97. It was slightly more than annoying.

With Agent, you can have your character move across the screen, point to a drop-down box, programmatically open the drop-down box and allow the character to speak to you about the options presented. I created a guided tour of my software and let Peedy, the parrot, point to boxes, fill in text boxes, change screens and generally walk the end user through the entire process of creating a new record in the database.

I found that using Agent saved me from having to write many tedious pages of documentation that detailed the steps necessary to create, save, and modify new records. It was also fun to develop. It allowed my creative side to participate in a positive and beneficial way. Creativity is pre-programmed in most developers and is a key component of what makes them successful. Creativity can and should be considered when developing your documentation, depending on the standards and expectations of your company.

The only feedback I received about my MS Agent experiment was that *someone had too much time on their hands* and it was never taken seriously, at least in part because of the comical looking character. It wasn't a lot of extra work to build, but it did require me to learn some new coding techniques. It was a pleasure when a person in our department was to be trained. I told them to take the guided tour. Perhaps Microsoft was ahead of their time and with a more respectable character, this type of technology could still become mainstream one day.

I recently built a computer for my dad as a 50th wedding anniversary gift. I documented some notes marked *Important PC Notes PLEASE Read* and left a shortcut on the desktop. I also created an audio file that documented the features and use of the computer. I had to ask him if he looked at my notes, but he offered to tell me that he took the case and computer audio tour.

These are just a few examples of alternative ways to document. It is this humble writer's opinion that new ways to document are underutilized and underestimated for their simplicity and potential impact in today's corporate environment.

The elusive be all and end all documentation software package has yet to be developed; there are a number of useful documentation tools that are designed for specific documentation tasks.

Copyright ©2006 CNET Networks, Inc. All rights reserved.
For more downloads and a free TechRepublic membership, please visit http://downloads.techrepublic.com.

8. Do the documentation yourself if possible

The best person to document is the builder. After all, who knows the system better than the system builder?

If you are the system builder, you are likely a crack programmer. But just mention the word *document* to a programmer and you will be given that "you've got to be kidding" look. If forced, programmers will document their work, or at least make an attempt to create something that will pass as documentation. I know. I have seen it all too often and even been guilty of it myself.

And that is a real shame because a programmer with good documentation skills is a valuable asset to the company. What will your manager remember at performance review time if another person had to do the documentation for your project? My guess is it won't be that you deserve a promotion, raise or bonus.

While not exactly fun, documentation can be rewarding when done correctly. Not only will you have a better overall project to present to your customer, you will also greatly reduce the future support time that you will have to provide. You can also reduce the amount of support and maintenance time for the help desk.

When working at CSC, I was given the opportunity to be the project lead for the design and creation of our global reporting system and infrastructure. I got to see first hand the other side of documentation. We had a very good programmer in the group who was doing Crystal Reports API work and custom function building. It was obvious to me that his knowledge was unique to him and needed to be shared with the rest of the team and what better way to do that then to properly document his work. I wasn't entirely successful in getting him to explain his work to the point that another person could step in and pick it up. He did list and explain the function names, how to use them, how they worked and what they accomplished and that was very helpful to the other members of the team.

There seems to be an unwritten rule in the Realm of the Code that programming skills are inversely proportional to the amount of documentation programmers have to do.

The second greatest compliment ever paid to me in my career was when I had to give a presentation to our global technical support team. I had to create and present documentation on how to build a reporting server. One of our database administrators was a chap from England who had to sit in on the presentation. He looked at the *How to Build a Reporting Sever* document and, to paraphrase, commented how good the documentation was and that he should be able to build a reporting server using my documentation. Statements like that make all of the hard work worth it. And it wasn't a compliment for the main project work - it was for the documentation.

9. Coordinate the development of the End User Documentation with the Internal/System Documentation

You can cut your documentation time if you build your User Documentation at the same time you write the System Documentation. You can share some of the information between the two and reduce missing information. Even if you don't want to or it is inappropriate to share information between the documents, you can benefit from topics in one document that will prompt you to include additional documentation in the other.

10. Follow department or corporate documentation guidelines

Create and follow standard formats and guidelines. This will help to insure that important information is not excluded and allow for easier reading by system users.

The one time I had a dedicated documentation expert work with me to document my system was at Hughes Aircraft Company. The results were excellent. The format was the department standard and the results were better than I would have done. It required a lot of time and effort to get those results. The documentation expert needed access to my beta system and access to me so that I could answer questions. This is more expensive and not all companies have the resources to allocate to professional documentation, but the results can be excellent if the system builder can verify that important information is not misinterpreted or left out of the end product.

I was very fortunate that I had a former engineer who was also excellent at writing documentation. He understood what the system was designed and built to do and he filled in the blanks by actually using the system and discovering for himself how it worked. You may not be so fortunate.

Copyright ©2006 CNET Networks, Inc. All rights reserved.
For more downloads and a free TechRepublic membership, please visit http://downloads.techrepublic.com.

In this day and age of global marketing, sales and support, documentation should also follow country or regional standards. I am often irritated reading the manual for a piece of electronic gear made in China that is all too often difficult to translate. It is written in Chinglish and for some sentences I have to pause and try to comprehend. I usually just make a mental Scooby Doo sound and move on to the rest of the manual.

Were English speaking documentation pros to learn (they never do) and write in Chinese, I imagine their Englese would sound the same to the Chinese speaking people. Find and use a professional translator to make the documentation understandable so that important information is not lost in translation.

I should also state the obvious. Your documentation should be free of misspelling and grammatical errors. Always use a spell-checker to find errors. I never ceased to be amazed at how many obvious spelling errors I have made and simply missed on a reread.

Documentation developer

If I haven't already convinced you that creating good documentation is good for you and your employer, take comfort in this fact - creating good documentation is not just a menial task. When you document your work you are now also a documentation *developer*. Hopefully these tips will help you to avoid those time consuming and disruptive questions that are bound to come your way and to your friendly help desk techs.

Authors note:

This is my first officially published document. I want to personally thank Sonja Thompson and Mark Kaelin for giving me this opportunity to share my thoughts with you. I'm not sure why Mark decided to offer me the opportunity to discuss such a daunting subject as documentation for my first article. Perhaps he figured that if I could write an interesting article about documentation, I could write about almost anything! I am glad he did though and my thanks to him and Tech Republic are more than I can convey with the tools of this trade.

A gentle reader has become a gentle writer.

Alan Norton – June 26, 2007

Additional references:

- *"...the four most important attributes about documentation include its content, maintenance, availability and use of examples."* Page 58 of http://www.site.uottawa.ca/~tcl/gradtheses/aforward/aforward_thesis.doc
- *The importance of documentation:* http://searchsmb.techtarget.com/originalContent/0,289142,sid44_gci1251087,00.html

Copyright ©2006 CNET Networks, Inc. All rights reserved.
For more downloads and a free TechRepublic membership, please visit http://downloads.techrepublic.com.

Additional resources

- TechRepublic's Downloads RSS Feed XML
- Sign up for TechRepublic's Downloads Weekly Update newsletter
- Sign up for TechRepublic's Software/Web Development NetNote newsletter
- Check out all of TechRepublic's free newsletters

Version history

Version: 1.0
Published: July 13, 2007

Tell us what you think

TechRepublic downloads are designed to help you get your job done as painlessly and effectively as possible. Because we're continually looking for ways to improve the usefulness of these tools, we need your feedback. Please take a minute to drop us a line and tell us how well this download worked for you and offer your suggestions for improvement.

Thanks!

—The TechRepublic Downloads Team

Copyright ©2006 CNET Networks, Inc. All rights reserved.
For more downloads and a free TechRepublic membership, please visit http://downloads.techrepublic.com.